Amendments to the Claims

1. (Currently Amended) A method for processing input from a command line interface, wherein the input comprises a macro, the method comprising:

replacing the macro with an executable command of the command line interface; and,

executing the command independent of compilation.

2. (Previously Presented) The method of claim 1, further comprising prompting a user to identify an executable command that is to replace the macro, wherein the replacing step further comprises replacing the macro with the identified executable command.

3. (Previously Presented) The method of claim 1, further comprising: prompting a user to identify a function that is to be used to generate the executable command to replace the macro; and calling the identified function to generate said executable command, wherein the replacing step further comprises replacing the macro with the generated command.

4. (Previously Presented) The method of claim 1, wherein the step of replacing further comprises calling a script engine to generate script to replace the macro with the executable command.

5. (Previously Presented) The method of claim 1, further comprising: prompting a user to identify a script that is to be used to generate the executable command to replace the macro; calling a script engine to execute the identified script to generate said executable command, wherein the replacing step further comprises replacing the macro with the generated command.

6. (Previously Presented) The method of claim 1, further comprising calling a function to generate an executable command to replace the macro, wherein the replacing step further comprises replacing the macro with the generated command.

7. (Original) The method of claim 3, wherein the function is called from a run-time library.

8. (Original) A computer-readable medium having stored thereon computer-executable instructions for performing the method of claim 1.

9. (Currently Amended) A method for processing a batch file comprising at least one macro, the method comprising:

parsing the batch file to locate text representing the macro;

expanding the macro into an executable command of a command line interface; and

executing the batch file, including the command, independent of compilation.

10. (Previously Presented) The method of claim 9, wherein the expanding step further comprises: in a first pass through the batch file, prompting the user to identify a function to be used to generate an executable command; replacing the macro with a second macro representing the identified function; in a second pass through the batch file, using the second macro to invoke the represented function and generate an executable command and replace the macro with the generated command.

11. (Previously Presented) The method of claim 9, wherein the expanding step further comprises: in a first pass through the batch file, locating a function identified by the macro; using the identified function to generate a second macro representing a second function; in a second pass through the batch file, using the second macro to invoke the second function and generate an executable command; and replacing the second macro with the generated command.

12. (Previously Presented) The method of claim 9, further comprising: prompting the user to input data for expanding the macro; reading a field in the macro to determine the type of data that is to be received from the user; and receiving the user input, wherein the step of expanding the macro is based on the determined type of data.

13. (Original) The method of claim 12, wherein, if the determined data type is a filename, providing a means for allowing the user to browse available files and select a file to be used to expand the macro.

14. (Original) A computer-readable medium having stored thereon computer-executable instructions for performing the method of claim 9.

15. (Currently Amended) A system for processing command line input, the system comprising:

a command line interface ~~for receiving the command line input~~ comprising a set of executable commands; and

a command line processor for, at least:

parsing the command line input[[,]];

identifying one or more macros within the input[[,]];

expanding the one or more macros into at least one executable command of the command line interface ~~commands,~~; and,

executing the commands independent of compilation.

16. (Original) The system of claim 15, further comprising a plug-in module for defining at least one of the macros, wherein the plug-in module is accessible by the command line processor.

17. (Original) The system of claim 15, further comprising a run-time library having functions that are executable by the command line processor to replace at least one of the macros with a line of text.

18. (Original) The system of claim 15, further comprising a run-time library having functions that are executable by the command line processor to replace at least one of the macros with another macro.

19. (Original) The system of claim 15, further comprising: a scripting engine invokable by the command line processor; and a computer-readable medium having stored thereon a script that is executable by the scripting engine to replace at least one of the one or more macros with a line of text when the scripting engine is invoked by the command line processor.

20. (Original) The system of claim 15, further comprising a computer-readable medium having stored thereon a text file having one or more lines of commands, wherein at least one of the lines of commands includes at least one of the one or more macros.

21. (Original) The system of claim 20, further comprising a means for reading the text file.

22. (New) The method of claim 1, wherein the command line interface comprises an operating system prompt.

23. (New) The method of claim 1, wherein the command line interface comprises a disk operating system (DOS) prompt.

24. (New) The method of claim 1, wherein replacing the macro with the executable command comprises loading a command line interface plug-in.

25. (New) The method of claim 1, wherein the macro comprises a dynamic-linked library macro specifying, at least:

a dynamic-linked library; and

at least one function within the dynamic-linked library.

26. (New) The method of claim 1, wherein the macro comprises a dialog box macro specifying, at least, a dialog box type.